MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

STRUČNI STUDIJ RAČUNARSTVO

IVAN MAGDIĆ

RAZVOJ ANDROID APLIKACIJE “POV- POINT OF VIEW - GRAD” UPOTREBOM GOOGLE MAPS

ZAVRŠNI RAD

ČAKOVEC, 2016.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU

STRUČNI STUDIJ RAČUNARSTVO

IVAN MAGDIĆ

RAZVOJ ANDROID APLIKACIJE “POV- POINT OF VIEW - GRAD” UPOTREBOM GOOGLE MAPS

DEVELOPING ANDROID APPLICATION “POV- POINT OF VIEW - CITY” USING GOOGLE MAPS

ZAVRŠNI RAD

Mentor:

mr.sc. Bruno Trstenjak, v. pred.

ČAKOVEC, 2016.

SAŽETAK

Cilj završnog rada je izrada mobilne aplikacije za prikaz lokacija u gradovima upotrebom Google Maps servisa. U radu je opisan proces razvoja mobilne aplikacije, kao i popratni web servis za unos i administraciju podatka o POV točkama.

Praktični dio projekta obuhvaća razvoj Android aplikacije koja omogućuje korisnicimapregled unesenih točaka. Unošenje točaka se obavlja pomoću web modula unutar Android aplikacije. Cilj Android aplikacije je olakšati i pomoći korisnicima u snalaženju i pregledu zanimljivih točaka odabranog grada. Aplikacija je strukturirana u više izbornika i aktivnosti za pregled podataka. Aplikacija podržava razne načine pregleda POV-a te filtriranje informacija prema definiranim uvjetima.Aplikacija također omogućuje korisniku filtriranje i pretraživanje gradova te pripadajućih POV točaka. U razvoju aplikacije korišten je Google Maps servis i Android Studio program. Pri razvoju aplikacije unutar Android Studio-a korištene su dvije vanjske biblioteke: Volley i Picasso. Volley je korišten za slanje podataka između Android i web aplikacije, a *Picasso za rad sa slikama i njihovo dohvaćanje sa web aplikacije po nazivu.Za potrebe ove mobilne aplikacije razvijen je web modul kojim se unose, uređuju i pregledavaju POV točke. Svi podaci o registriranim POV točkama i korisnicima zapisani su u MySQL bazi koja je postavljena na web poslužitelju u okruženju web modula. Web aplikacija je pisana u PHP programskom jeziku uz korištenje HTML, CSS i JavaScript jezika za izradu korisničkog sučelja na web-u. Web aplikacija se nalazi na web poslužitelju gdje je dio za unos točke javan, a dio za administriranje je ograničen s korisničkim računom sa administrativnim ovlastima. Sam razvoj web modula ostvaren je pomoću Komodo IDE razvojnog okruženjadok je MySQL Workbench alatkorišten za dizajniranje baze podataka.U završnom radu detaljno je objašnjen način upotrebe Google Maps servisa i način se ti podaci prikazuju, korištene razvojne okoline i tehnologije, strukturuaplikacije te najvažnije aktivnosti, metode i module aplikacije. Tijekom razvoja aplikacije provedena su testiranja kvalitete mobilne aplikacijena emulatoru i uređajima koji se razlikuju po specifikacijama i veličini zaslona što je prikazano u odlomku 7.*

Ključne riječi: Android aplikacija, webaplikacija, Android Studio,Google Maps, Point of View

SADRŽAJ

1. UVOD

U današnje vrijeme Android aplikacije postale su svakodnevica. Cilj ovog završnog rada je izrada Android mobilne aplikacije koja omogućuje pristup informacijama o interesantnim lokacijama u nekom gradu. Lociranje tih točaka na mapi obavlja se putem internetske veze i Google servisa.Interesne točke na mapi nazvane su Point of View i sadrže informacije kao što su naziv točke, ulica, kratki opis točke te sliku točke.

Unatoč već sličnim aplikacijama i integriranim točkama u Google Maps sučelju, ova aplikacija nudi korisniku proširivanje baze točaka u raznim gradovima. Problem kod sličnih aplikacija je vremenski proces dodavanja, upravljanja i uređivanja točaka. Taj problem je riješen u ovoj aplikaciji na način da se svakom korisniku nudi opcija dodavanje nove točke na temelju trenutne pozicije, unosom osnovnih informacija o točki i odabirom slike lokacije na koju se točka odnosi.

Android aplikacija razvijena je u Java programskom jeziku. Osim mobilne aplikacije razvijen je i web modul za upravljanje podacima o POV. U razvoju aplikacije korištenje Android Studio te Komodo IDE.

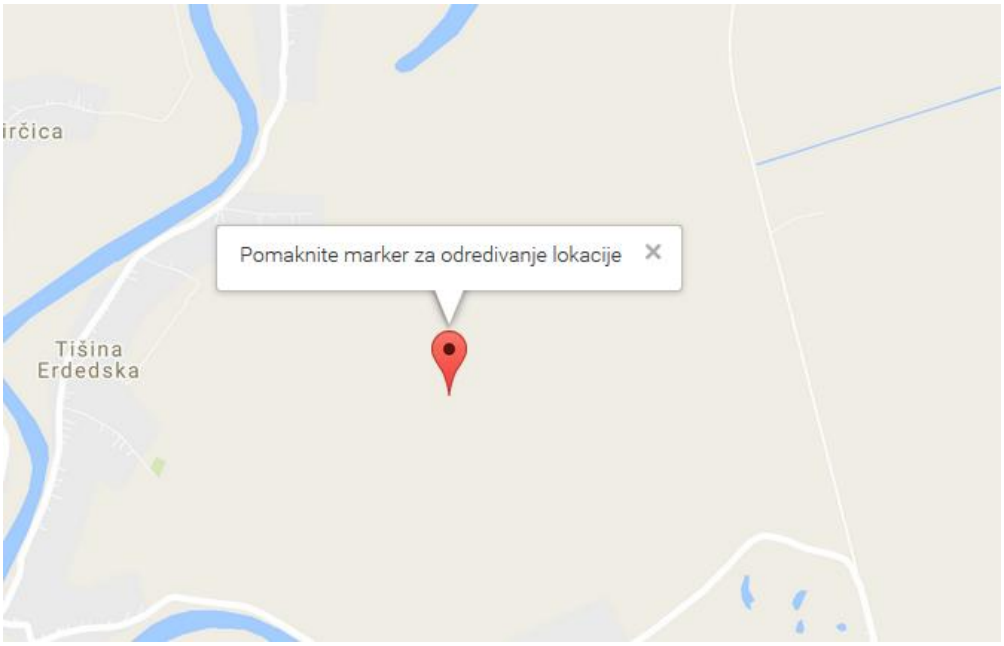
Pohrana, unos, uređivanje i pregledavanje točaka je odrađeno pomoću web aplikacije koja je razvijena u Komodo IDE-u. Korišteni su razni programski jezici, akomunikacija između mobilne aplikacije i web modula ostvarena je slanjempodataka u JSON formatu zapisa.

2. CILJ ZAVRŠNOG RADA

Cilj ovog završnog rada je izrada Android aplikacije za lakše snalaženje korisnicima u stranim gradovima. U ostvarivanju zadanog cilja potrebno je koristiti Google Maps framework za implementaciju servisa pristupa lokacijama, te izraditi mobilnu aplikaciju koja s web modulom čini sustav evidentiranja POV točaka. Također, cilj je da aplikacija bude što jednostavnija za upotrebu. Osim izrade Android aplikacije, cilj je proširiti vlastito znanje i kompetencije u području razvoja mobilnih aplikacija potrebnih za tržište rada.

3. POINT OF VIEW (POV)

Point of View u ovoj Android aplikaciji je gledište korisnika koje je definirano kutem, razinom zuma (*engl. Zoom level*), brojem točkama i mapom.Za prikazivanje POV točaka u aplikacija koristi se Google Mapsservis koji nam omogućuje prikazivanje Google-ove mape. Svaka točka se sastoji od dva dijela, a to su marker i prozor s informacijama. Marker čine dvije stavke koje se prenašaju u opisu točke, a to su geografska širina (*engl. Latitude*) i geografska dužina (*engl. Longitude*). Drugi dio točke je prozor sinformacijama (*engl. Info Window*), koji sadrži informacije: naziv točke, ulicu, opis, dodatan opis, grad i naziv slike. Za prikaz prozora sinformacijama korisnik treba kliknuti na željeni marker. Slika1. prikazuje marker i prozor s informacijama.

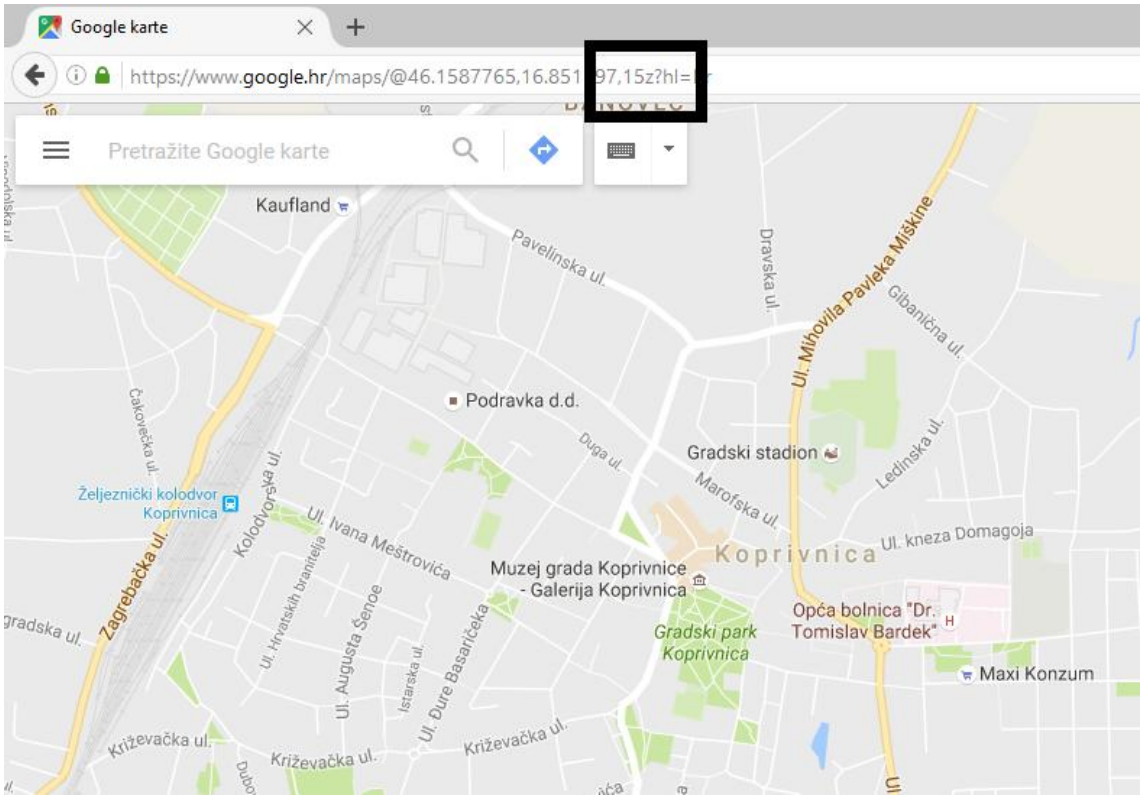


Slika 1. Prikaz markera i prozora s informacijama

Izvor: autor završnog rada

Svaka točka predstavlja POV(*engl. Point of View*). Aplikacija koristi razinu mape koja prikazuje putokaze, odnosno putokaz mapa (*engl. Roadmap).*POV je u aplikaciji moguće odrediti na sljedeće načine: korisnik klizi prstom na mapi ili ako korisnik dva

puta klikne na zaslon što poveća razinu zuma. Na početku, aplikacija prikazuje popis gradova. Odabirom jednog grada iz popisa, u aplikaciji se prikazuje novi zaslon gdje korisnikvidi zadnju dodanu točku. Ta točka je centrirana na zaslonu, a razina zumiranja je uvijek 15. Svakom promjenom grada ili interakcije na mapi korisniku se mijenja POV. Slika 2. prikazuje razinu zumiranja kada korisnik klikne na grad na Google mapi. Na slici 2. je prikazana Google mapa, te gdje i kako možemo doznati razinu zumiranja na Google mapi pomoću URL-a[1].Trenutna razina u ovom primjeru je 15 što je označeno na slici 2. sa „15z“.



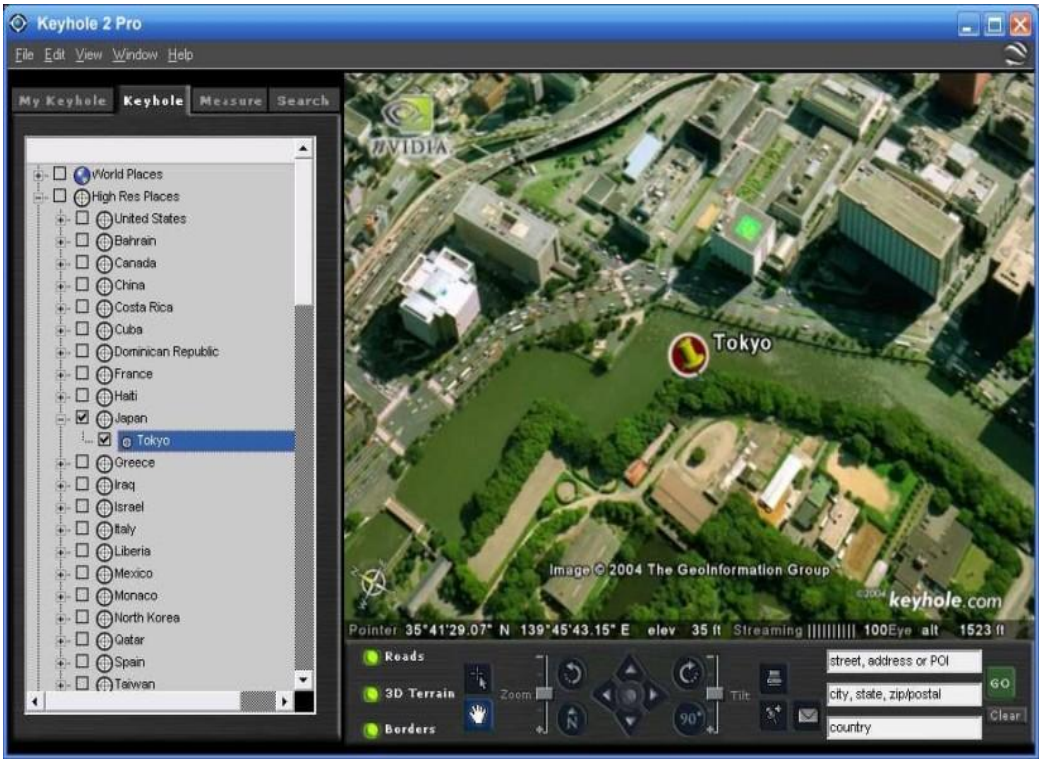
Slika 2. Prikaz razine zumiranja u URL-u

Izvor: autor završnog rada

[1]URL (*engl. Uniform Resource Locator*) - usklađeni lokator sadržaja

4. GOOGLE MAPS SERVIS

Google Maps je Google-ov besplatni servis digitalnih mapa. Omogućuje korisniku pregled satelitskih snimaka, plan kretanja, traženje određenih mjesta, 360 stupnjeva pregled određene lokacije i još mnogo toga.Servis je razvijen 2004 godine od strane tvrtke Keyhole, a godinu dana kasnije Google kupuje tu tvrtku.Servisu dodaje vizualizaciju i objavljuje svoj Google Maps servis javnosti[1].Slika 3. prikazuje zaslon aplikacije tvrtke Keyhole prije nego što je Google kupio prava.



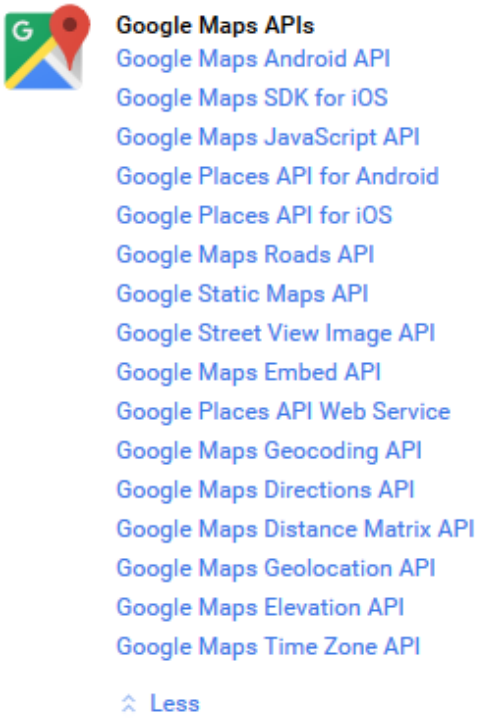
Slika 3.Prikaz aplikacije tvrtke Keyhole

Izvor:
http://www.syix.com/elmer/4%20Screenshots/Keyhole%20Pro%20(820%20x%20545).jpg (9.9.2016.)

Google Maps je objavljen na internetu u veljači 2005. godine, a dva mjeseca poslije implementira nove mogućnosti: pregled satelitskih snimki (*engl. satellite views*) i korištenje navigacije [2]. Godinu dana kasnije, 2006. Google Maps se pojavljuje na mobilnim uređajima na Europskom tržištu gdje pokriva sljedeće države: Ujedinjeno Kraljevstvo, Francusku, Njemačku, Italiju i Španjolsku [2]. Danas, Google Maps nudi

navigaciju u više od 190 država. Godine 2005., Google objavljuje svoj prvi Google Maps API za razvoj aplikacija baziranih na tom Google servisu [2].

Google API(*engl. Application Programming Interfaces*) je skup sučelja za razvoj aplikacija koje koriste neku od Google usluga. Google je s objavom tog API-a ponudio sljedeće usluge: Google Mail, Google Search, Google Translate i Google Maps. Slika 4. prikazuje sve Google Maps API-e koje je moguće razvijati i koristiti.

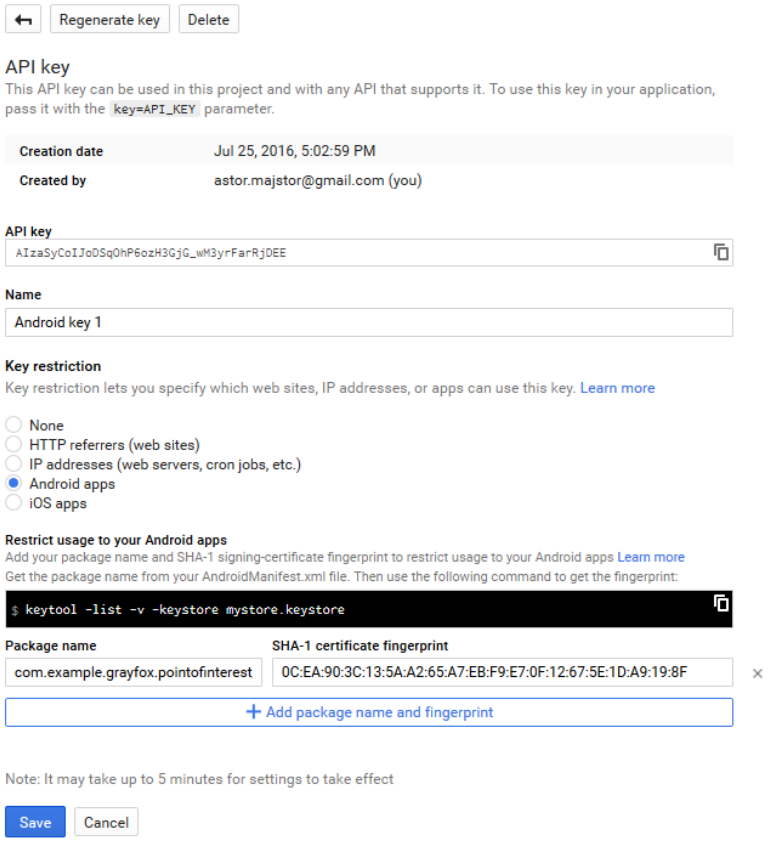


Slika 4. Prikaz svih Google Maps API

Izvor: https://console.developers.google.com/apis/library?project=electric-attic-138323 (7.9.2016.)

U razvoju aplikacije korišten je Google Maps Android API, a pri razvoju web aplikacije korišten je Google Maps JavaScript API. Tim uslugama aplikacija je omogućila prikaz mape gradova u raznim rezolucijama i detaljima. Mape se mogu prikazati na dva načina: digitalni način i satelitski način. Digitalne mape sadrže vektorske slike, a satelitske mape sadrže snimke koje su rasterskog tipa [3]. Mape prikazuju sva naseljena područja s informacijama o zemljopisnim lokacijama.

Za razvoj ovakvih ili sličnih aplikacija potrebno je dobiti certifikat od strane Google-a. Dobivanje certifikata ostvaruje se u nekoliko koraka.Prvo je potrebno napraviti Google račun (*engl. Account*), poznato kao Google Mail ili Gmail, te zatražiti odobrenje za korištenje njihovog sučelja u vezi razvoja. U ovom radu korišten je sljedeći API: ”Google Maps Android API v3". Naravno, da bi Google znao jesuli korištene njihove mape potrebno je poslati šifru/oznaku računala koja se naziva SHA-1. Slika 5. prikazuje dio web stranice za unos SHA-1 šifre.



Slika 5. Prikaz unosa SHA-1 šifre u web formu na Google stranici

Izvor: https://console.developers.google.com/apis/credentials/key/0?project=electric-attic-138323 (9.9.2016.)

Podatak o SHA-1 oznaci dobiva seklikom na "Start" ikonu, pod "traži" se upiše: "cmd". Otvori nam se crni prozor te u njemu napišemo sljedeću naredbu : "set path=C:/ProgramFiles/Java/jdk1.7.0_11/bin". S time je stavljena putanja sljedeće

naredbe: "keytool –list –v –keystore C:/Users/.android/debug.keystore –alias androiddebugkey –storepass android –keypass android". Potvrdom unosa na zaslonu se prikažu podaci o računalu. Između njih nalazi se traženi SHA-1 certifikat. Taj podatak potrebnoje napisati na Google stranici za razvijače (*engl. developers*) da se uvjere jesu li traženi uvjeti zadovoljeni. Ukoliko su uvjeti zadovoljeni, Google će generiratiAPI ključ koji je potreban za korištenje Google servisa u budućoj aplikaciji. API ključ u ovom radu se unosio unutar razvojne okoline Android Studio IDE. Izradom novog projekta, unutar AndroidManifest.xml dokumenta naveden je sljedeći kod:

Kod 1. Povezivanje Android aplikacije i Google Maps API-a

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.grayfox.pointofinterest_rh">
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
<application
        android:name=".AppController"
        android:allowBackup="true"
        android:icon="@drawable/logo"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
<activity android:name=".MainActivity" />
<meta-data
        //definiranje API ključa
         android:name="com.google.android.geo.API_KEY"
         android:value="AIzaSyCoIJoDSqOhP6ozH3GjG_wM3yrFarRjDEE
" />
<activity
            android:name=".MapsActivity"
            android:label="@string/title_activity_maps" />
<activity android:name=".ZupanijeActivity">
</activity>
<activity android:name=".AdapterZupanija" />
<activity android:name=".GradoviActivity"
            android:windowSoftInputMode="stateHidden" />
<activity android:name=".AdapterGradova" />
<activity android:name=".DetaljiActivity" />
<activity android:name=".IndexActivity">
…
```

5. RAZVOJNA OKOLINA

Prilikom razvoja završnog rada korištene su sljedeće razvojne okoline i tehnologije: Android Studio IDE, Komodo IDE za razvoj web modula, MySQL Workbench za razvoj MySQL baze podataka. U daljnjem tekstu objašnjene su neke od razvojnih okolina i tehnologija.

5.1. Android Studio IDE

Android Studio je službeno integrirano razvojno okruženje za razvoj mobilnih aplikacija u skladu s Android operacijskim sustavom [4]. Android operacijski sustav, odnosno Google Android je prvi otvoreni operacijski sustav za mobilne uređaje od Google tvrtke. Android Studio je sličan Eclipse ADT-u jer su oba bazirana na programskom jeziku Java. U razvoju mobilne aplikacije korišten je Android Studio 2.1 verzija programa tvrtke InteliJ IDEA. Android Studio sadrži editor i SDK Manager [2]sa svim alatima koji su potrebni za razvoj i testiranje aplikacije. Budući da se Android Studio temelji na Java programskom jeziku, potrebno je imati instaliran najnoviji Java Development Kit (JDK) ukoliko se već ne nalazi u paketu sa samim IDE-om. Na slici 6. je prikazan logo Android Studio IDE programa.

Slika 6. Prikaz logo-a Android Studio IDE programa

Izvor: autor završnog rada

Alat je nadograđen s dvije vanjske biblioteke: Volly i Picasso. Biblioteka „Volley“se koristi za komunikaciju Android aplikacije s web aplikacijom. Druga korištena biblioteka je „Picasso“, koja daje podršku za rad sa slikama i njihovo dohvaćanje s web

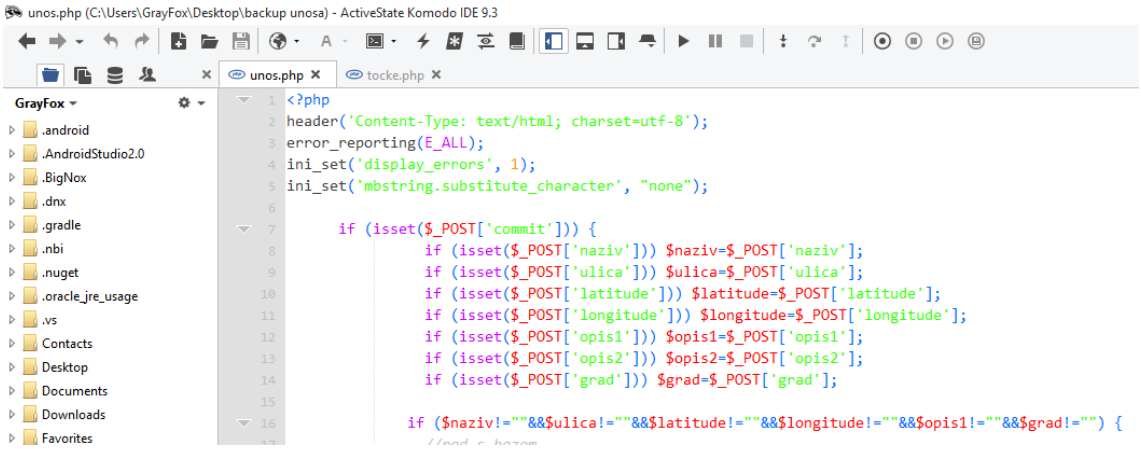
[2] SDK Manager (*engl. Software Development Kit Manager*) – alat koji služi za kontrolu instaliranih paketa

aplikacije po nazivu. Osim dohvaćanja slika, „Picasso“ biblioteka omogućava prilagodbu slike zavisno o zaslonu korisnika.

5.2. Komodo IDE

Komodo IDE je integrirano razvojno okruženje koje podržava pisanje programa u raznim programskim jezicima[5].Komodo je korišten tijekom razvoja web modula aplikacije.U razvojusukorišteni PHP i JavaScript jezici. Komodo IDE se koristio u kreiranju web stranice za unos, pregled i provjeru podataka u PHP programskom jeziku.U razvoju, PHP kod nadopunjen je HTML, JavaScript i CSS naredbama.

Skripte razvijene pomoću PHP jezika korite se za unos i kontrolu podataka na razini web aplikacije. U njima su ugrađene funkcije za pristup MySQL bazi te pohranu podataka o POV točkama. HTML zajedno s CSS za vizualni dizajn web stranica.Dizajn i funkcionalnost web stranici podržane su od strane Bootstrap biblioteke. JavaScript je imao ulogu za prikaz interaktivne mape na zaslonu pri unosu nove točke na web aplikaciji. Slika 7. prikazujezaslon Komodo IDE.



Slika 7. Prikaz zaslona Komodo IDE

Izvor: autor završnog rada

6. STRUKTURA APLIKACIJE

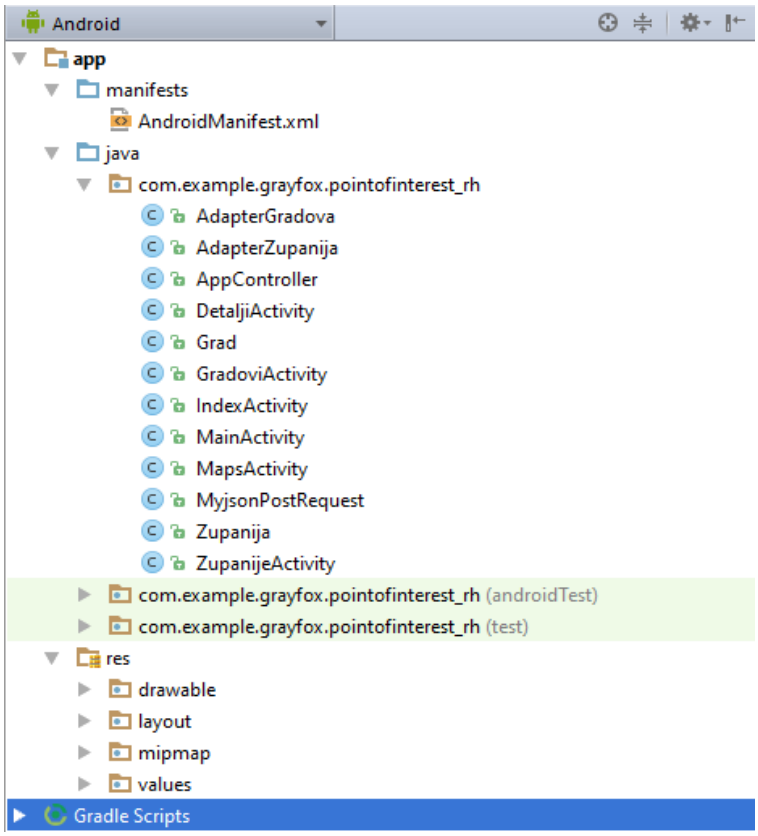
Struktura Android mobilne aplikacije sastoji se od tri osnovne mape i jedne skripte za pokretanje aplikacije.. Nazivi mapa su: „manifests“, „java“ i „res“. Mapa „manifests“ sadrži „AndroidManifests.xml“dokument. U njemu su pohranjene sve informacije o aplikaciji. Neke od zapisanih informacija o projektu su: naziv aplikacije, popis svih aktivnosti, dozvole koje aplikacija može izvršiti i pristupiti, ikona aplikacije, popis korištenih biblioteka itd.

Kod 2. Primjer koda iz AndroidManifests.xml datoteke

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.grayfox.pointofinterest_rh">
//deklariranje dozvola koje aplikacija može izvršiti i pristupit
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
<application
        //informacije o aplikaciji (logo, naziv aplikacije,tema)
        android:name=".AppController"
        android:allowBackup="true"
        android:icon="@drawable/logo"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
       //popis svih aktivnosti
<activity android:name=".MainActivity" />
<meta-data
        android:name="com.google.android.geo.API_KEY"
        android:value="AIzaSyCoIJoDSqOhP6ozH3GjG_wM3yrFarRjDEE
/>
<activity
        android:name=".MapsActivity"
         android:label="@string/title_activity_maps" />
<activity android:name=".ZupanijeActivity">
</activity>
<activity android:name=".AdapterZupanija" />
<activity android:name=".GradoviActivity"
        android:windowSoftInputMode="stateHidden" />
<activity android:name=".AdapterGradova" />
<activity android:name=".DetaljiActivity" />
<activity android:name=".IndexActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category
android:name="android.intent.category.LAUNCHER" />
```

```
</intent-filter>
</activity>
</application>
</manifest>
```

Druga mapa se naziva „java“ iu njoj se nalazi izvorni kod mobilne aplikacije, sve klase i aktivnosti razvijene u java programskom jeziku. Više o aktivnostima se može pročitati u daljnjem tektsu u odlomku 6.2.. Posljednja mapa „res“, sadrži sva korisnička sučelja u podmapi „layout“ i sve slike koje se koristeu aplikaciji. One su zapisane u podmapi „drawable“. U mapi su također zapisane informacije o tekstualnim konstantama i nazivima u podmapi „values“. Slika 8. prikazuje strukturu Android aplikacije.



Slika 8. Prikaz strukture projekta

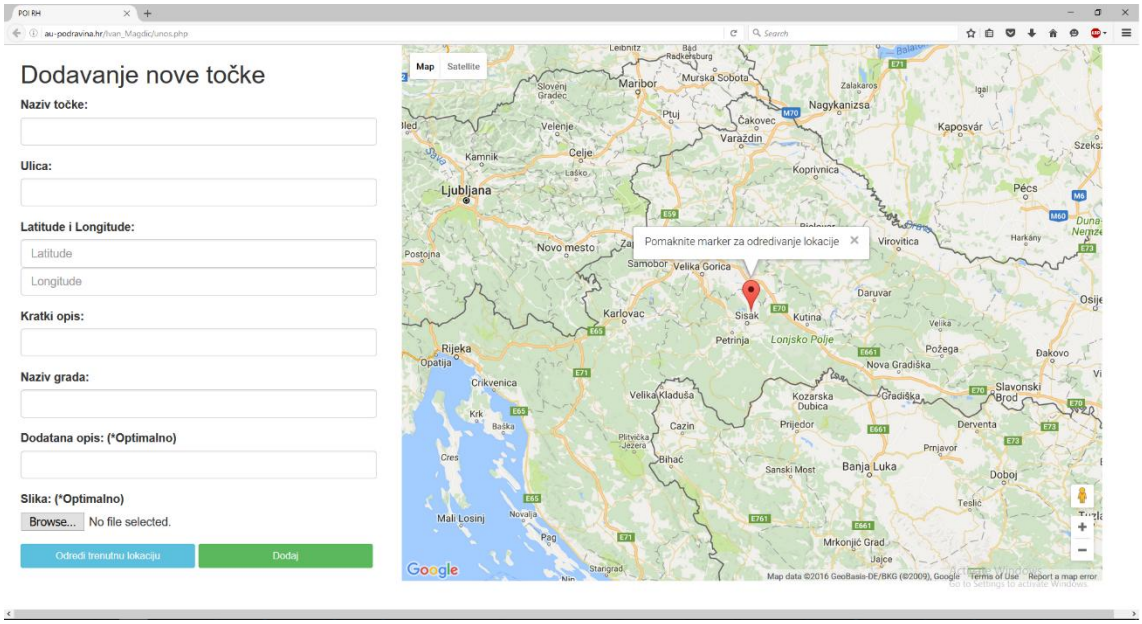
Izvor: autor završnog rada

6.1. Moduli aplikacije

Zbog kompleksnosti aplikacije, ova mobilna aplikacija organizirana je u tri međusobno povezana modula. Prvi modul je web modul odnosno web aplikacija, drugi modul je modul za slanje i primanje podataka te zadnji modul koji je namjenjen pregledavanju

gradova i njihovih POV-a koji je sastavni dio mobilne aplikacije.Zadnji modul sastavni je dio mobilne aplikacije.

Web modul se sastoji od elementa za unos i administriranje POV točaka. Unos novih POV točaka je otvoren za pristup, može ga koristiti svaki korisnik aplikacije. Administrativni element ograničen jekorisničkim računom sadministrativnim pravima. Slika 9. prikazuje zaslon web preglednika za unos POV točaka.



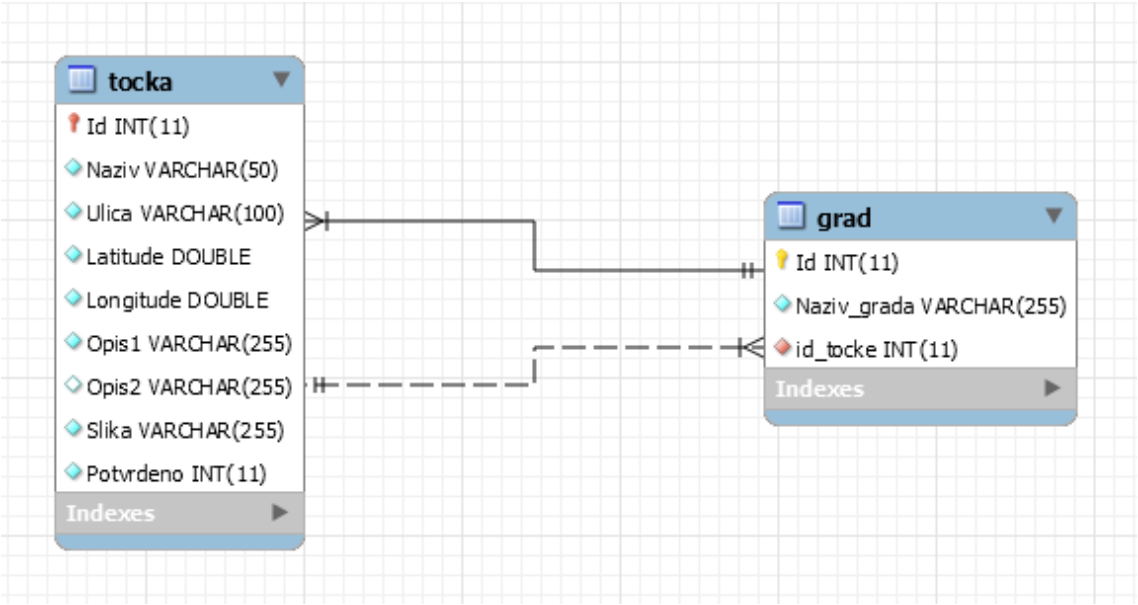
Slika 9. Prikaz zaslona elementa za unos točke

Izvor: autor završnog rada

Element za unos sastoji se od dva dijela: forme za unos informacija koje definiraju POV točku i Google mape.Za korištenjeGoogle mape potrebno je također registrirati web aplikaciju i dobiti ključ za aktiviranje licence. Forma i Google mapa su stilizirane s Bootstrap paketom. Bootstrap omogućuje razvoj responzivnih web stranica koje se prilagođavaju različitim uređajima i njihovim zaslonima [6].

Kada korisnik ispuni podatke u formi i pritisne gumb „Dodaj“ pokreće se validacija podataka.Ukoliko nisu popunjena sva obavezna polja, korisniku se na zaslonu pojavi poruka s upozorenjem. Podaci iz forme zapisuju se u MySQL bazu podataka. Slika 10. prikazuje dijagram baze, točnije ERD(engl. *entity relationshipdiagram*).

Baza se sastoji se od dvije tablice: grad i tocka. Svaka tablica ima atribute koji su određenog tipa podatka. U svakoj tablici se nalazi atribut Id koji je ključ za međusobno povezivanje tablica.



Slika 10. ERD dijagram baze podataka

Izvor: autor završnog rada

Modul za slanje i primanje podataka u Android aplikaciji sastoji se od nekoliko aktivnosti: IndexActivity i GradoviActivity. U IndexActivity aktivnostikorištena je GET metoda za dohvaćanje podataka sa servera, odnosno baze podataka. Sljedeći kodprikazuje diometodeslanja upita prema web aplikaciji:

Kod 3. Dio koda iz metode „Start“

```
//deklariranje i popunjavanje varijable za upit prema web app.
…
String url = "http://au-
podravina.hr/Ivan_Magdic/grad.php?Zupanija=sve";
//Inicijaliziranje objekta „Zahtjev“ sa parametrima
MyjsonPostRequest Zahtjev = new
MyjsonPostRequest(Request.Method.GET, url, new
Response.Listener<JSONArray>() {
…
```

U kodu je prikazana varijabla url čija vrijednost se šalje web aplikaciji. Sadržaj varijable se prosljeđuje u objekt „Zahtjev“ kao inicijalna vrijednost. Prvi parametar inicijalizacije objekta je tip zahtjeva metode, drugi parametar je podatak koji želimo poslati web aplikaciji, a treći parametar je metoda koja čeka povratnu informaciju web aplikacije. Povratna informacija u ovom slučaju su svi gradovi zapisani u bazi. Zaprimljeni podaci su u JSON formatu i spremaju se u listu, a samo oblikovanje i slanje JSON podataka se odvija na web aplikaciji. U sljedećem odlomku je opširnije objašnjena metoda i princip slanja i primanja podataka.

Zadnji modul aplikacije je modul za pregledavanje gradova i njihovih POV-a. U mobilnoj aplikaciji za prikaz podataka koriste se „GradoviActivity“ i„MapsActivity“ aktivnosti. GradoviActivity aktivnost u obliku liste prikazuje popis gradova čije podatke je web aplikacija poslala.MapsActivity aktivnost na isti način prikazuje podatke samo što ona prikazuje POV točke na mapi. Prikazivanje podataka GradoviActivity aktivnosti se odrađuje u metodi „onCreate“. Sljedeći kod prikazuje metodu „onCreate“:

Kod 4. metoda „onCreate“ GradoviActivity aktivnosti

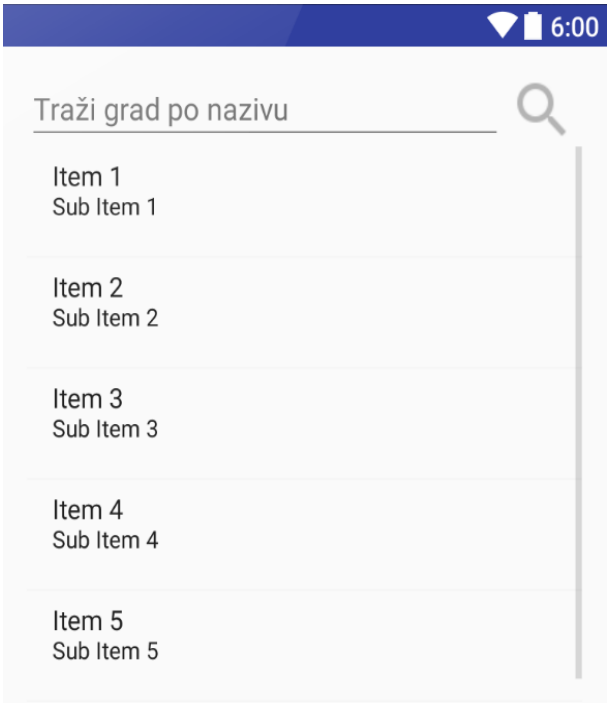
```
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_gradovi);
myList = (ListView) findViewById(R.id.listViewGradova);
gradovi =getIntent().getParcelableArrayListExtra("gradovi");

myAdapter = new AdapterGradova(this,
R.layout.activity_adapter_gradova, gradovi);
myList.setAdapter(myAdapter);

myList.setOnItemClickListener((new
AdapterView.OnItemClickListener() {
@Override
public void onItemClick(AdapterView<?> parent, View
view, int position, long id) {
// selected item
String selected = ((TextView)
view.findViewById(R.id.textViewAdapterGrad)).getText().toString(
);
     Intent pokreniGoogleMaps = new
Intent(getApplicationContext(), MapsActivity.class);
     pokreniGoogleMaps.putExtra("grad", selected);
     startActivity(pokreniGoogleMaps);
        }
```

```
        }
        ));
}
```

Metoda „onCreate“ se izvršava odmah pri pokretanju aktivnosti. U njoj se provodi inicijalizacija podataka o registriranim gradovima te se formira popis gradova u obliku liste. Također, u metodi je definiran način interakcije korisnika s popisom gradova. Klikom na grad pokreće se sljedeća aktivnost, a to je MapsActivity koja je objašnjena opširnije u sljedećem odlomku.

6.2. Najvažnije aktivnosti i metode u mobilnoj aplikaciji

Aktivnost u Android aplikacijama predstavlja programsku nit koja omogućuje interakciju korisnika s djelom aplikacije. U mobilnoj aplikaciji najvažnije su sljedeće aktivnosti: IndexActivity, GradoviActivity, MapsActivity i DetaljiActivity.



Slika 11. Prikaz korisničkog sučelja aktivnosti „activity_gradovi.xml“

Izvor: autor završnog rada

Android aplikacije većinom imaju najmanje jednu aktivnost ili imaju niz aktivnosti koje su povezane međusobno. Tok izvršavanja aplikacije preusmjerava korisnika na ostale aktivnosti pomoću grafičkog korisničkog sučelja (*engl. graphical user interface*).

Najčešće se aktivnost sastoji iz dva dijela. Prvi dio je XML datoteka koja definira grafički izgled zaslona. Drugi dio je program, odnosno programska logika koja je vezana za tu aktivnost. Slika 11. prikazuje korisničko sučelje "GradoviActivity" aktivnosti iz Android aplikacije.

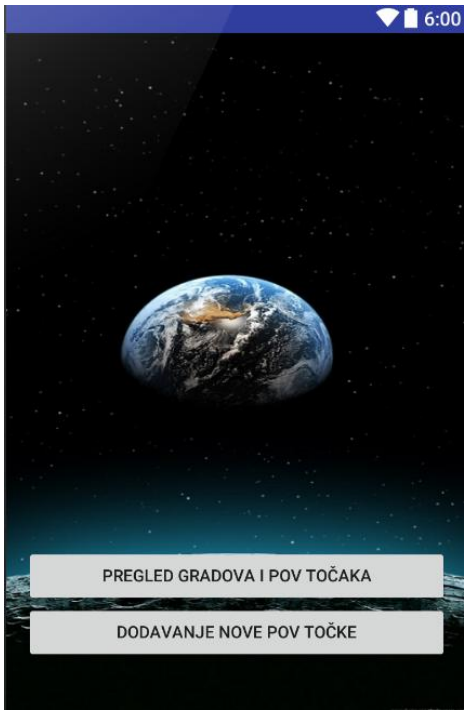
Sljedeći kod prikazuje „activity_gradovi.xml" datoteke:

Kod 5. activity_gradovi.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res
/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"

tools:context="com.example.grayfox.pointofinterest_rh.GradoviAct
ivity">
<ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/listViewGradova"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/editTextTraziGradove" />
<ImageView
        android:layout_width="50dp"
        android:layout_height="wrap_content"
        android:id="@+id/imageViewTraziGrad"
        android:src="@drawable/places_ic_search"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:layout_above="@+id/listViewGradova"
        android:onClick="TraziGrad" />

<EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/editTextTraziGradove"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:hint="Traži grad po nazivu"
        android:layout_toLeftOf="@+id/imageViewTraziGrad"
        android:layout_toStartOf="@+id/imageViewTraziGrad" />
</RelativeLayout>
```

U prikazanom kodu može se vidjeti da je korisničko sučelje definirano u tekstualnom obliku. Ovim kodom se implementirala lista gradova (ListView), gumb za pretraživanje (ImageView) i polje za pretraživanje (EditText). Svi navedeni elementi raspoređeni su u „RelativeLayout“ elementu koji ih prikazuje u povezanom odnosu.

IndexActivity je aktivnost koja prikazuje sadržaj početnog zaslona. Slika 12. prikazuje grafički izgled IndexActivity aktivnosti.



Slika 12. Prikaz korisničkog sučelja IndexActivity aktivnosti

Izvor: autor završnog rada

IndexActivity korisniku nudi dvije opcije: pokretanje ActivityGradovi aktivnosti klikom na gumb “PREGLED GRADOVA I POV TOČAKA“ ili pokretanje web preglednika klikom na gumb „DODAVANJE NOVE POV TOČKE“. Ukoliko je korisnikodabrao „PREGLED GRADOVA I POV TOČAKA“, pokreće se metoda „Start“. U slučaju odabira druge opcijepoziva se metoda „Tocke“.

Kod 6. Metoda „Tocke“

```
public void Tocke (View v){
//pokretanje web preglednika
    Intent browserIntent = new Intent(Intent.ACTION_VIEW,
Uri.parse("http://au-
podravina.hr/Ivan_Magdic/unos_mobitel.php"));
    startActivity(browserIntent);
}
```

Metoda „Start“je jedna od važnijih metoda.U njoj su implementirane naredbe za komunikaciju sa web aplikacijom i dohvaćanjepodataka o svim gradovima koji su zapisani u sustavu. Sljedeći kod prikazuje „Start“ metodu:

Kod 7. Metoda „Start“

```
public void Start(View v) {
String url = "http://au-
podravina.hr/Ivan_Magdic/grad.php?Zupanija=sve";
MyjsonPostRequest Zahtjev = new
MyjsonPostRequest(Request.Method.GET, url, new
Response.Listener<JSONArray>() {
        @Override
        public void onResponse(JSONArray response) {
            //'sluša' povratnu informaciju web aplikacije
            try {
                for (int i = 0; i < response.length(); i++)
                {
                    JSONObject grad = (JSONObject)
response.get(i);
                    String name = grad.getString("naziv");
                    Grad g = new Grad();
                    g.setNaziv_grada(name);
                    gradovi.add(g);
                }
                Intent objIndent = new
Intent(getApplicationContext(), GradoviActivity.class);
                objIndent.putParcelableArrayListExtra("gradovi",
gradovi);
                startActivity(objIndent);
                gradovi.clear();
            } catch (JSONException e) {
                e.printStackTrace();
                Toast.makeText(getApplicationContext(),
                "Error: " + e.getMessage(),
                Toast.LENGTH_LONG).show();
            }
        }
    }, new Response.ErrorListener() {
```

```
//metoda se izvrši ukoliko ima grešaka pri dohvaćanju JSON
podataka
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getApplicationContext(),
            "Error: nesto ne valja" + error.getMessage(),
            Toast.LENGTH_LONG).show();
        }
    });
//objekt se stavlja na red izvršavanja
AppController.getInstance().addToRequestQueue(Zahtjev);
}
```

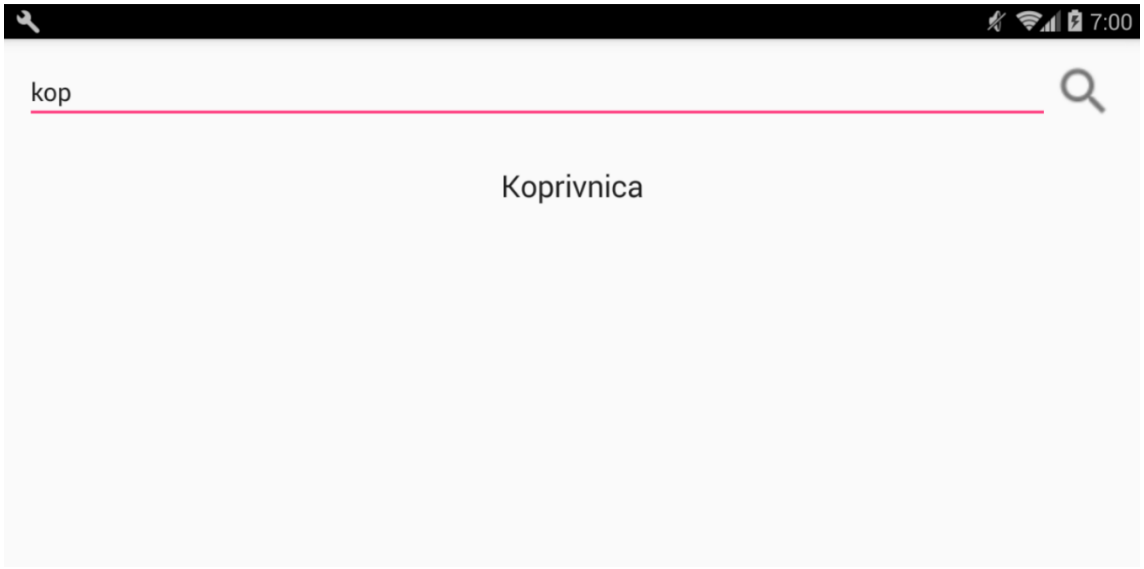
Nakon što metoda dohvati sve gradove, sprema ih u listu.Lista podataka prosljeđuje se u aktivnost GradoviActivity.U aktivnosti GradoviActivity implementirana je metoda za pretraživanje gradova po nazivu. Sljedeći kod prikazuje metodu za traženje grada po nazivu:

Kod 8. Metoda „TraziGrad“

```
public void TraziGrad(View v){
    myList = (ListView) findViewById(R.id.listViewGradova);
    myList.setAdapter(null);
    ArrayList<Grad> gradovi_temp = new ArrayList<>();
    int brojac_gradova = 0;

    EditText ET =
(EditText)findViewById(R.id.editTextTraziGradove);
    String Naziv_grada = ET.getText().toString();

    if(Naziv_grada.equals("")){ // provjera dali je polje prazno
    myAdapter = new AdapterGradova(this,
R.layout.activity_adapter_gradova, gradovi);
    myList.setAdapter(myAdapter);
    } else { // ako nije,naredba provjerava ima li polje broj u
sebi
        if (
Naziv_grada.contains("1") || Naziv_grada.contains("2") ||
Naziv_grada.contains("3") || Naziv_grada.contains("4") ||
Naziv_grada.contains("5") || Naziv_grada.contains("6") ||
Naziv_grada.contains("7") || Naziv_grada.contains("8") ||
Naziv_grada.contains("9"))
{
//ukoliko ima broj,naredba ispisuje poruku korisniku
Toast.makeText(getApplicationContext(), "Nije dovoljeno
upisivanje brojeva", Toast.LENGTH_LONG).show();

        } else { //ukoliko polje nema broj izvršava se slj. kod
            for (int a = 0; a < gradovi.size(); a++)
            {
```

```
                Grad grad = gradovi.get(a);
                String naziv = grad.getNaziv_grada();
                if
(naziv.toLowerCase().contains(Naziv_grada.toLowerCase().toString
()))
                {//popunjavanje privremene liste sa objektom
grad ako se podudara naziv grada i uneseni pojam
                    gradovi_temp.add(grad);
                    brojac_gradova++;
                }
            }
            if (brojac_gradova == 0)
            { //ukoliko nema rezultata pri traženju grada po
pojmu ispiše se poruka korisniku
            Toast.makeText(getApplicationContext(), "Nije nađen
ni jedan grad pod navedenim nazivom", Toast.LENGTH_LONG).show();
            myAdapter = new AdapterGradova(this,
R.layout.activity_adapter_gradova, gradovi);
            myList.setAdapter(myAdapter);
            } else { //ukoliko je nađen grad po pretraživanom
pojmu prikazuje se u listi
            myAdapter = new AdapterGradova(this,
R.layout.activity_adapter_gradova, gradovi_temp);
            myList.setAdapter(myAdapter);
            }
        }
    }
}
```

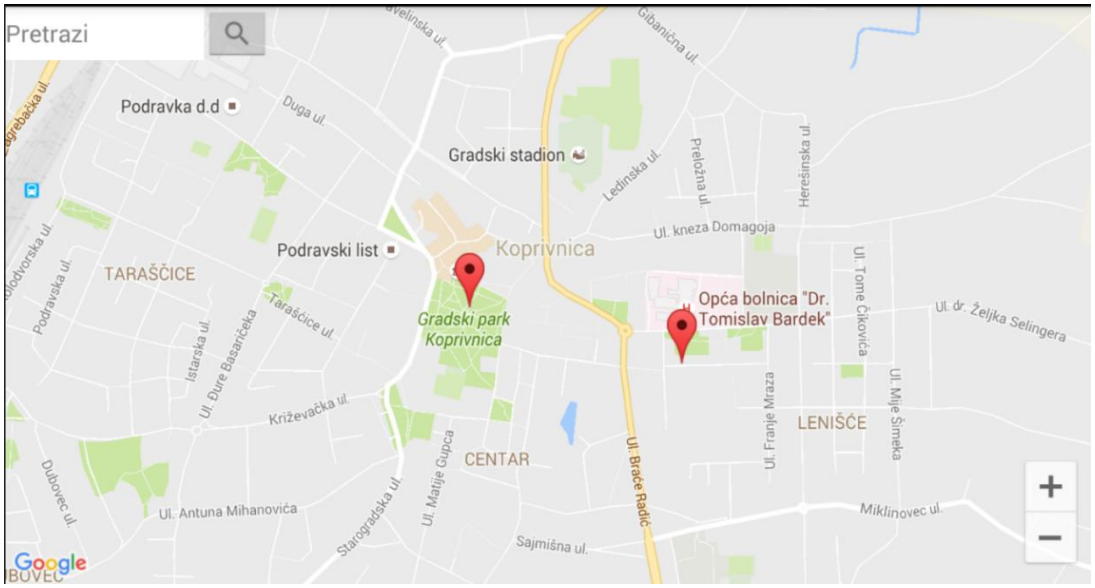
Prilikom upisa riječi u polje za traženje i klikom na gumb sa slikom povećala, pokreće se „TraziGrad“ metoda. Metoda prvo inicijalizira i isprazni listu,napravi privremenu listu u koju se spremaju objekti te dohvati tekst iz polja gdje smo upisali pojam. Metoda filtrira popis gradova na temelju upisanog teksta u polju za pretraživanje. U metodi je ugrađen validator koji ne dozvoljava upis brojeva iz sigurnosnih razloga. Ukoliko je sve u redu s upisanim tekstom pretraživanja, metoda popunjava privremenu listu s gradovima koji u sebi sadrži tekst kojeg je korisnik unio u polje.Slika 13. prikazuje primjer pretraživanja grada po nazivu.



Slika 13. Prikaz zaslona aplikacijeprilikom traženja grada po unesenom pojmu

Izvor: autor završnog rada

Sljedeća važna aktivnost je MapsActivity.U njoj je implementirana metoda za prikaz POV točaka odabranog grada.Na slici 13. je prikazana lista sjednim gradom „Koprivnica“.Odabirom grada iz popisa pokreće se MapsActivity aktivnost. Aktivnost prikazuje sve POV točke povezane uz odabrani grad. Na slici 14. je prikazan izgled mape s istaknutim POV točkama nakon odabira grada.



Slika 14. Prikaz zaslona aplikacije prilikom odabira grada „Koprivnica“

Izvor: autor završnog rada

Sljedeći kodprikazuje diometode koja se izvšava s ciljem prikazivanja POV točke na mapi:

Kod 9. Dio koda metode „onMapReady“

```
//slanje upita prema php-u da primi sve tocke koje su povezeane
//i potvrdene sa gradom
MyjsonPostRequest Zahtjev = new
MyjsonPostRequest(Request.Method.GET, url, new
Response.Listener<JSONArray>() {
    @Override
    public void onResponse(JSONArray response) {
        try {
            for (int i = 0; i < response.length(); i++) {
               JSONObject grad = (JSONObject) response.get(i);
               String name = grad.getString("Naziv");
               String ulica = grad.getString("Ulica");
               Double Lat = grad.getDouble("Latitude");
               Double Long = grad.getDouble("Longitude");
               String opis1 = grad.getString("Opis1");
               String opis2 = grad.getString("Opis2");


//dodavanje markera na Google Mapi
LatLng LL = new LatLng(Lat,Long);
Marker marker = mMap.addMarker(new MarkerOptions()
       .position(LL)
       .title(name)
       .snippet(opis1 + "\n" +opis2 + "\n\n" + ulica)
       .icon(BitmapDescriptorFactory

.defaultMarker(BitmapDescriptorFactory.HUE_RED))
                );
        mMarkers.put(marker.getId(),marker);
        ListaMarkera.add(grad);
//prikazivanje zadnje unesene točke na mapi
mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(LL,15));
            }
        } catch (JSONException e) {
            e.printStackTrace();
            Toast.makeText(getApplicationContext(),
            "Error: " + e.getMessage(),
            Toast.LENGTH_LONG).show();
        }
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Toast.makeText(getApplicationContext(),
        "Error: nesto ne valja" + error.getMessage(),
```

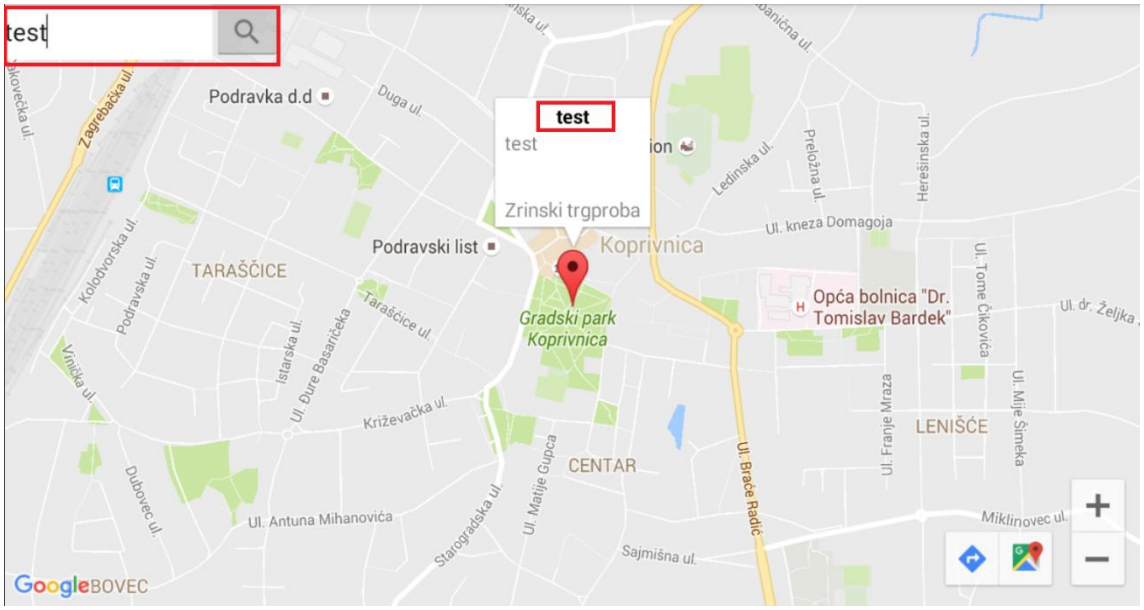
```
            Toast.LENGTH_LONG).show();
        }
});

AppController.getInstance().addToRequestQueue(Zahtjev);
```

U prikazanom kodumetoda zaprima proslijeđeni podatak iz prijašnje aktivnosti, što je u ovom slučaju naziv grada. Naziv grada se šalje web aplikaciji koja kao povratnu informaciju Android aplikaciji šalje sve točke i njezine atribute u JSON formatu zapisa.Kada Android aplikacija zaprimi podatke, oni se zapisuju u listu unutar MapsActivity-a. Zaprimljeni JSON podaci spremaju se u objekt „Marker“. Tako formirani podaci prikazuju se na mapi odabranog grada. Kod 10 prikazuje formiranje objekta „Marker“ sa JSON podacima:

Kod 10. Formiranje „Marker“ objekta

```
Marker marker = mMap.addMarker(new MarkerOptions()
            .position(LL)//geo. Širina i dužina točke
            .title(name)//naziv točke
            //kratki opis točke:
            .snippet(opis1 + "\n" +opis2 + "\n\n" + ulica)
            //ikona točke
            .icon(BitmapDescriptorFactory
            .defaultMarker(BitmapDescriptorFactory.HUE_RED))
);
```

Ovako formirani i pripremljeni podaci o POV točama omogućuje kasnije njihovo pretraživanje prema upisanom pojmu. MapsActivity aktivnostima mehanizam pretraživanja koji je identičan kao u aktivnosti GradoviActivity.Na mapi se prikazuju samo one POV točke koje sadrže traženi tekst u bilo kojem dijelu svoje informacije.Na slici 15. je prikazan zaslon mobilne aplikacije nakonfiltriranja POV točaka.



Slika 15. Prikaz zaslona aplikacije prilikom traženja točke po nazivu „test“

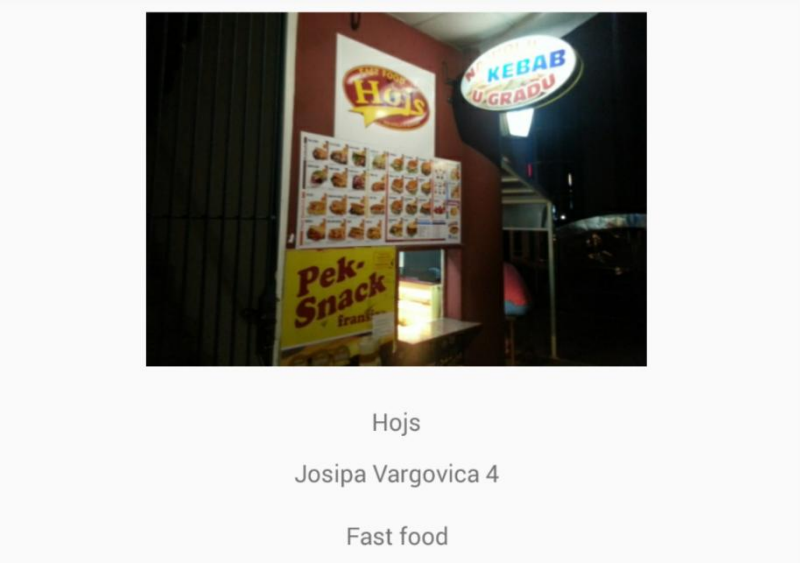
Izvor: autor završnog rada

Zadnja važna aktivnost je DetaljiActivity. Odabirom POV točke prikazuju se dodatne informacijekorisniku po cijelom zaslonu. Također, prikazuje se i slika koja je pridodana lokaciji. Aktivnost se pokreće pritiskom na prozor s informacijama.Kod 11. prikazuje metodu za praćenje interakcije korisnika i prozora s informacijama.

Kod 11. Metoda za pokretanje DetaljiActivity aktivnosti prilikom interakcije korisnika na prozor sinformacijama

```
mMap.setOnInfoWindowClickListener(new
GoogleMap.OnInfoWindowClickListener() {
    @Override //metoda se izvšava kada korisnik klikne ma prozor
sa informacijama određene točke
    public void onInfoWindowClick(Marker marker) {

        Marker mar = (Marker)mMarkers.get(marker.getId());
        String naziv_tocke = mar.getTitle();

        for(int i =0; i<ListaMarkera.size();i++){
            try {
            JSONObject grad = ListaMarkera.get(i);
            String naziv = grad.getString("Naziv");
            String opis1 = grad.getString("Opis1");
            String opis2 = grad.getString("Opis2");
            String ulica = grad.getString("Ulica");
            String slika = grad.getString("Slika");
```

```
            if (naziv.equals(naziv_tocke)){
//provjera kliknutog markera i POV točaka u listi , ako se
nalazi u //listi izvšava se slj. naredba
                Intent intent = new
Intent(getApplicationContext(),DetaljiActivity.class);
                    intent.putExtra("naziv",mar.getTitle());
                    intent.putExtra("ulica",ulica);
                    intent.putExtra("opis1",opis1);
                    intent.putExtra("opis2",opis2);
                    intent.putExtra("slika",slika);
                    startActivity(intent);
//pokretanje DetaljiActivity aktivnosti sa podacima klikne točke
                }
            } catch (JSONException e) {
                e.printStackTrace();
                Toast.makeText(getApplicationContext(),
                "Error: " + e.getMessage(),
                Toast.LENGTH_LONG).show();
            }
        }
    }
});
```



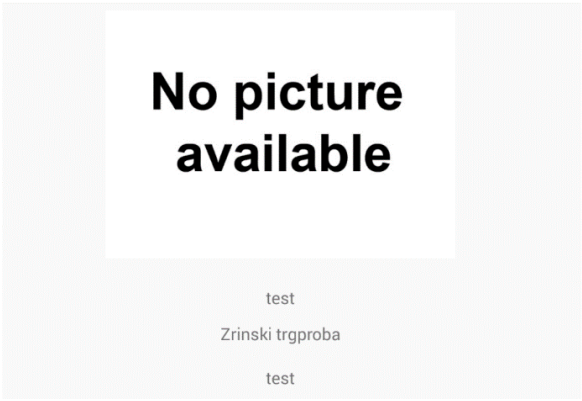
Slika 16. Prikaz zaslona aplikacije klikom korisnika na točku koja se naziva „Hojs“

Izvor: autor završnog rada

Aplikacija prvo prikuplja informacije o odabranoj POV točki. Zatim uspoređuje naziv pritisnute POV točke na mapi sa svim spremljenim POV točkama na mapi. Ukoliko

nađe podudarajuću POV točku, pročita određene podatke iz nje i proslijedi ih u sljedeću aktivnost DetaljiActivity. Prikaz zaslona DetaljiActivity aktivnosti prikazan je na slici 16.

Grafičko korisničko sučelje aktivnosti DetaljiActivity sastoji se od slike POV točke i dopunskih informacijama: nazivom točke, naziva ulice u gradu i opisa lokacije. Ukoliko kod definiranja nove POV točke nije određena pripadajuća slika, na zaslonu se prikazuje oznaka o tome kao prikazuje slika 17.



Slika 17. Prikaz zaslona aplikacije kada točka nema definiranu sliku pri unosu

Izvor: autor završnog rada

7. TESTIRANJE APLIKACIJE

Android aplikacija razvijena je dapodržavarazličite uređaje koji koriste različite verzije softvera (*engl. Software*). Na taj način proširena je lista uređaja na kojima se može izvršavati ova aplikacija. Softver verzije kod Android operacijskog sustava su poznate pod nazivom „API platforme“ (*engl. API framework*). Svaka API platforma pripada određenom API nivou (*engl. API level*)[7].Slika 18. prikazuje sve API nivoe koji su do danas razvijeni (6.9.2016.).

Platform Version	API Level	VERSION_CODE	Notes
Android 7.0	24	N	Platform Highlights
Android 6.0	23	M	Platform Highlights
Android 5.1	22	LOLLIPOP_MR1	Platform Highlights
Android 5.0	21	LOLLIPOP	
Android 4.4W	20	KITKAT_WATCH	KitKat for Wearables Only
Android 4.4	19	KITKAT	Platform Highlights
Android 4.3	18	JELLY_BEAN_MR2	Platform Highlights
Android 4.2, 4.2.2	17	JELLY_BEAN_MR1	Platform Highlights
Android 4.1, 4.1.1	16	JELLY_BEAN	Platform Highlights
Android 4.0.3, 4.0.4	15	ICE_CREAM_SANDWICH_MR1	Platform Highlights
Android 4.0, 4.0.1, 4.0.2	14	ICE_CREAM_SANDWICH	
Android 3.2	13	HONEYCOMB_MR2	
Android 3.1.x	12	HONEYCOMB_MR1	Platform Highlights
Android 3.0.x	11	HONEYCOMB	Platform Highlights
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1	Platform Highlights
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD	
Android 2.2.x	8	FROYO	Platform Highlights
Android 2.1.x	7	ECLAIR_MR1	Platform Highlights
Android 2.0.1	6	ECLAIR_0_1	
Android 2.0	5	ECLAIR	
Android 1.6	4	DONUT	Platform Highlights
Android 1.5	3	CUPCAKE	Platform Highlights
Android 1.1	2	BASE_1_1	
Android 1.0	1	BASE	

Slika 18. Prikaz svih API nivoa

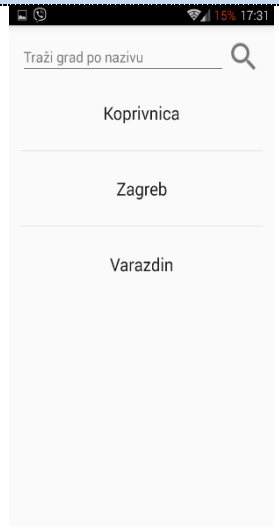
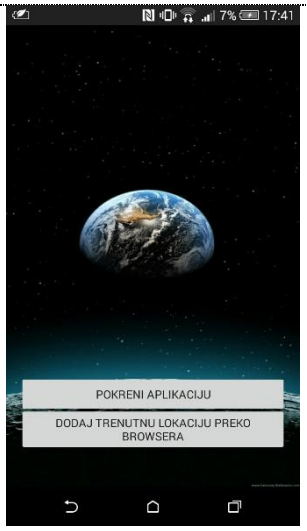
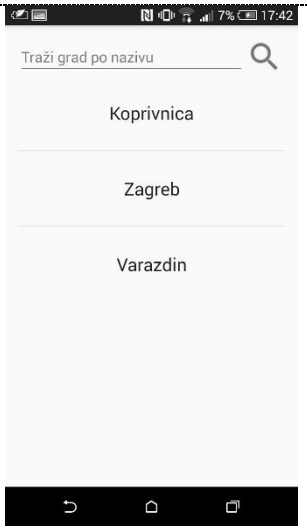
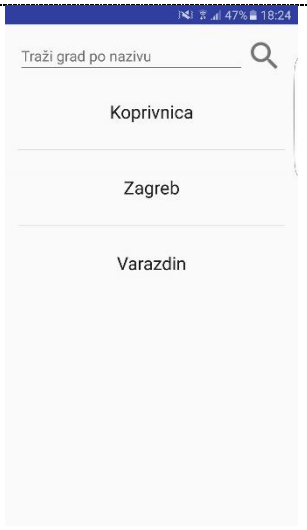
Izvor: https://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels (9.9.2016.)

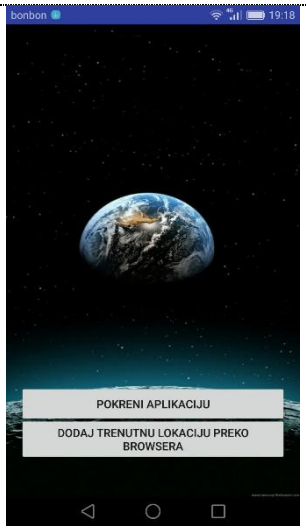
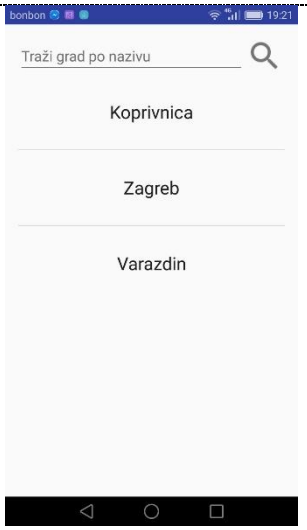
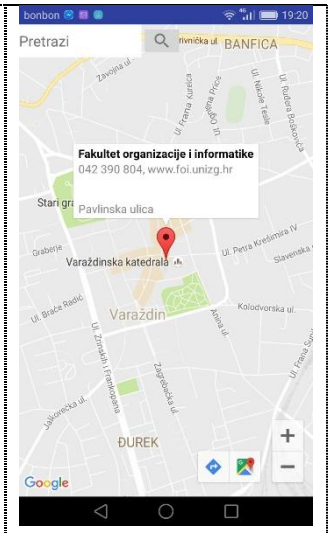
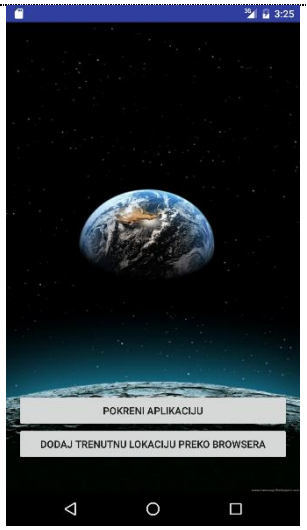
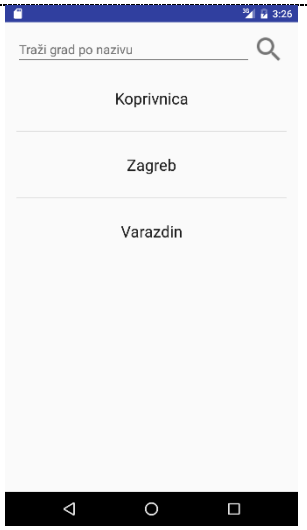
Aplikacija je testirana na emulatoru koji se izvršavao na API nivou 23, te na uređajima: Samsung Galaxy S3 i S6 EDGE, HTC Desire 620 i Huawei P8.

Samsung Galaxy S3 ima zaslon dimenzija 720x1280 piksela (*engl. Pixels*)[8]. Instaliran je API nivo 18, točnije 4.3 platform verzija. Samsung Galaxy S6 EDGE ima zaslon dimenzija 1440 x 2560 piksela (*engl. Pixels*)[9]. Instaliran je API nivo 21, 5.0.2 platform verzija. HTC Desire 620 je dizajniran sa 720 x 1280 piksela zaslonom i izvršava se na 19 API nivou tj. 4.4.3 verziji platforme[10]. Posljednji uređaj je Huawei P8 te on ima zaslon 1080 x 1920 piksela i API nivo 19, točnije 4.4.2 verziju platforme[11].

Evaluacija kvalitete rada aplikacije pokazala je da se aplikacija uspješno izvršava. Tablica u nastavku prikazuje zaslone navedenih uređaja na kojima je aplikacija testirana. Na uređajima je testirana instalacija aplikacije, pokretanje aplikacije, unos POV točke, filtriranje i prikazivanje POV točaka. Pokretanje i instalacija aplikacije ovise o specifikacijama uređaja gdje glavnu ulogu imaju procesor i memorija uređaja. Pozornost pri testiranju bila je na grafičkom sučelju Android aplikacije, određivanju lokacije korisnika u web modulu, prikazu podataka u listi gradova i prikazu POV točaka na mapi. Grafičko sučelje na svim testiranim uređajima je prilagođeno veličini zaslona. Određivanje lokacije korisnika na uređajima radi, ali postoje odstupanja do 50 metara od trenutne lokacije. Stupac „Slika zaslona 2“ i „Slika zaslona 3“ u tablici 1. prikazuje zaslone ekrana uređaja gdje su zaprimljeni određeni podaci. Stupac „Slika zaslona 2“ prikazuje listu s podacima gradova, a stupac „Slika zaslona 3 “ prikazuje POV točku grada. Najbolji uređaj na kojem su podaci bili prikazani je Samsung Galaxy S6 EDGE zbog veličine zaslona ekrana. Slanje podataka između web i Android aplikacije odrađeno je podjednako na svim uređajima.

Tablica 1. Pregled uređaja korištenih kod testiranja aplikacije

Uređaj	Slika zaslona 1	Slika zaslona 2	Slika zaslona 3
Samsung Galaxy S3			
HTC Desire 620			
Samsung Galaxy S6 EDGE			

Huawei P8			
Emulator			

Izvor: autor završnog rada

8. ZAKLJUČAK

Ovim završnim radom prikazana je izrada Android mobilne aplikacije. Izgradnjom Android aplikacije nastojao se postići najjednostavniji prikaz točaka koristeći se Google servisom. Aplikacija sadrži jednostavno grafičko sučelje za prikaz POV točaka, pretraživanje te dodavanje novih.Jednostavnost sučelja osigurala je jednostavnost u korištenju aplikacijom s ciljem da se korisnik što manje optereti.

Cilj postavljen ovim završnim radom je ostvaren. Izrađena je Android aplikacija koja omogućuje korisnicima pregled, pretraživanje i dodavanje POV točaka za odabrani grad. Struktura i koncept aplikacije omogućava njenu upotrebu za bilo koji grad.

Aplikacija je razvijena uz pomoć Android Studio IDE-akoji se je pokazao kao kvalitetan i profesionalni alat.Temelj aplikacije je Google mapa uz module za prikazivanje podataka, web modula za unos i administriranje točaka i modula za slanje i primanje podataka. Web modul je izrađen u Komodo IDE-u uz korištenje PHP, JavaScript, HTML i CSS jezike. Komodo IDE se koristio u kreiranju web stranice za unos, pregled i provjeru podataka u PHP programskom jeziku.U razvoju, PHP kod nadopunjen je HTML, JavaScript i CSS naredbama.HTML zajedno s CSS korišten je za vizualni dizajn web stranica, a JavaScript je korišten za prikaz mape na web stranici.

Izradom završnog rada stekao sam novo znanje, unaprijedio postojeće i stekao mnogo iskustva. Najviše iskustva i znanja sam stekao pri radu s Android aplikacijom odnosno Java programskim jezikom uz korištenje posebnih elemenata i biblioteka potrebne za razvoj Android aplikacije. Unapređenje aplikacije je uvijek moguće i biti će ostvareno u novijim verzijama gdje će biti dodane nove ideje uz unapređenje postojećih.

9. POPIS LITERATURE

[1] Youtube video.
https://www.youtube.com/watch?v=HMBJ2Hu0NLw (3.9.2016.)

[2] About Google.
https://www.google.co.uk/about/company/history/#2005(3.9.2016.)

[3] John Schaeffer\Juniper GIS:Spatial Analyst and Raster Analysis.
http://www.junipergis.com/files/2312/5945/8037/Spatial_Analyst_and_Raster_Analysis.pdf (6.9.2016.)

[4] Što je Android Studi IDE.
http://www.tutorialspoint.com/android/android_pdf_version.htm (9.9.2016.)

[5] Komodo IDE.
http://community.activestate.com/files/komodo-ide-6.1.2.pdf (6.9.2016.)

[6] Bootstrap.
http://www.whoishostingthis.com/resources/bootstrap/ (6.9.2016.)

[7] API Levels http://developer.android.com/guide/topics/manifest/uses-sdk element.html#ApiLevels(7.9.2016.)

[8] GSMARENA specifikacije HTC Desire 620 mobilnog uređaja
http://www.gsmarena.com/htc_desire_620-6830.php(6.9.2016.)

[9] GSMARENA specifikacije Samsung Galaxy S3 mobilnog uređaja
http://www.gsmarena.com/samsung_i9300_galaxy_s_iii-4238.php
(6.9.2016.)

[10] GSMARENA specifikacije Samsung Galaxy S6 EDGE mobilnog uređaja
http://www.gsmarena.com/samsung_galaxy_s6_edge-7079.php (6.9.2016.)

[11] GSMARENA specifikacije Huawei P8 mobilnog uređaja
http://www.gsmarena.com/huawei_p8-7006.php (6.9.2016.)

PRILOZI

POPISSLIKA

POPISTABLICA